3 Computational considerations

- In what follows, we analyze the complexity of the Simplex algorithm more in detail
- For this purpose, we focus on the update process in each iteration of this procedure
- Clearly, since, up to now, the illustration of the algorithm was the primary aim, we updated the Tableau completely
- However, if we code the algorithm in some computer language this is not efficient
- Therefore, we introduce the so-called revised Simplex algorithm

3.1 The Revised Simplex Algorithm

- By considering the Simplex Algorithm generated above, it turns out that we have to update a complete *(m+1)x(n+1)* tableau throughout the calculation process
- Additional analyses, however, show that we can reduce this effort by keeping a significantly smaller *(m+1)x(m+1)* tableau
- Specifically, this is established by making use of the following scheme

Basic observations

- If we commence our calculations with the identity matrix E_m, we finally obtain the inverted matrix of A_B corresponding to a current basis B
- Thus, we start with a zero cost row as the objective function. This enables us to determine the corresponding dual solution
- Since c is given as a parameter, we can identify the relative costs by using a simple transformation
- At iteration l, we denote the considered *(m+1)x(m+1)* matrix as $CARRY^{(l)}$
- In addition, we commence the calculation with $CARRY^{(0)}$

Thus, we work with the following tableau

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0\ldots0 & 0 \\ \hline b_1 & 1 & 0 & 0\ldots0 & 0 \\ \ldots & 0 & 1 & 0\ldots0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ b_m & 0 & 0 & 0\ldots0 & 1 \end{array} = \left(\begin{array}{c|c} 0 & 0^m \\ \hline b & E_m \end{array}\right) = CARRY^{(0)}$$

$$\Rightarrow \left(\begin{array}{c|c} -Z_0 & -\pi^T \\ \hline A_B^{-1} \cdot b & A_B^{-1} \end{array}\right) = CARRY^{(l)}$$

Transformations

- Additionally, we keep track of the basis *B*, i.e., the current basic variables that form the current bfs
- In order to carry out the Primal Simplex Algorithm, the following steps are necessary

1. Generate the relative costs $\bar{c}_j = c_j - \pi^T \cdot a^j$ one at a time until we either find a negative - say for $j = s$ - value or we terminate with the cognition that the current solution is optimal already

Transformations

2. Column Generation:

Generate the column $\bar{a}^s = A_B^{-1} \cdot a^s \Rightarrow$ Generate the pivot element r with $\frac{\bar{b}_r}{\bar{a}_{r,s}} = \min\left\{\frac{\bar{b}_p}{\bar{a}_{p,s}} \mid p \in \{1,\ldots,m\} \wedge \bar{a}_{p,s} > 0\right\}$. Clearly, if r does not exist, the problem is unbounded.

3. Update $CARRY^{(l)}$ to obtain $CARRY^{(l+1)}$. By making use of vector $\bar{a}^s$, we are able to update A_B^{-1} and $\bar{b}$ accordingly.

4. Update the basis *B* accordingly. Specifically, we replace *B(r)* by *s*.

Column generation

- The second step introduces new columns, i.e., a new alternative in the tableau
- By making use of the inverted matrix of the current A_B, we can iteratively generate the columns of the original primal tableau
- In step 3, we apply all updating operations to the inverted matrix as well as to the left-hand side, i.e., to $\bar{b} = A_B^{-1} \cdot b$

Applied to the Two-Phase Method

- Note that **applying the Revised Simplex to the Two-Phase Method** comes along with several specifics we must attend to
- First of all, the initial solution coincides with the maximal usage of the *m* auxiliary variables (for every row one variable)
- Hence, the inverted matrix A_B^{-1} is E_m
- In order to commence with correct cost values for the column *j* that does not belong to the basis, we determine

$$\bar{c}_j = -\sum_{i=1}^{m} a_{ij} - \pi^T \cdot a^j$$

- For the second phase, we have to adjust the objective function row by using

$$-\pi^T = -c_B^T \cdot A_B^{-1}$$

The Revised Primal Simplex Algorithm I

In what follows, we assume that the right hand side *b* is positive. If this does not apply we have to use the two-phase method instead

1. Transform the primal problem into canonical form. Generate equations with the slack variables denoted as $x_1,\ldots,x_m$ and let $x_{m+1},\ldots,x_{m+n}$ be the structure variables. Obtain a minimization objective function.

2. Start with the feasible basic solution to the primal problem given by the slack variables: Obtain $\bar{b} = b, \pi^T = 0$ and $A_B^{-1} := E_m, B(1) = 1,\ldots, B(m) = m$

3. Search for a pivot column $s := \min\left\{ j \middle| \bar{c}_j = c_j - \pi^T a^j < 0 \right\}$
   If *s* exists:
   - Calculate $\bar{a}^s := A_B^{-1} \cdot a^s$.
   - If $\forall i = 1,\ldots,m : \bar{a}_{is} \le 0$, then terminate since the solution space is unbounded.
   - Pivot row r: $\lambda_0 := \min\left\{ \bar{b}_i / \bar{a}_{is} \middle| \forall i = 1,\ldots,m : \bar{a}_{is} > 0 \right\}$ that is an upper bound on x_s.

The Revised Primal Simplex Algorithm II

- Form the elementary matrix

$$P=\left(e^1,\ldots,\underbrace{p^r}_{\text{Column } r},\ldots,e^m\right) \text{ with } \left(p^r\right)^T=\left(p_{1r}:=-\frac{\bar{a}_{1s}}{\bar{a}_{rs}},\ldots,p_{rr}:=\frac{1}{\bar{a}_{rs}},\ldots,p_{mr}:=-\frac{\bar{a}_{ms}}{\bar{a}_{rs}}\right)^T$$

- Basis change (x_s enters the basis and $x_{B(r)}$ becomes a non-basic variable)
  Calculate $A_B^{-1}:=P\cdot A_B^{-1}$ and set B(r):=s
- Calculate $\bar{b}:=A_B^{-1}\cdot b, \pi^T:=c_B^T\cdot A_B^{-1}$
- Go to step 3.

Otherwise ($\bar{c}_j\geq 0\ \forall j=1,\ldots,n$): Termination. An optimal basic solution to the primal problem is found.

Example – Revised Simplex Method

Maximize $\begin{pmatrix}4 & 2 & 3\end{pmatrix}\cdot x$

$$\begin{pmatrix}4 & 1 & 2\\ 3 & 3 & 3\end{pmatrix}\cdot x\leq\begin{pmatrix}20\\ 30\end{pmatrix}, x\geq 0\Rightarrow$$

$$A=\begin{pmatrix}1 & 0 & 4 & 1 & 2\\ 0 & 1 & 3 & 3 & 3\end{pmatrix}, c=\begin{pmatrix}0 & 0 & -4 & -2 & -3\end{pmatrix}, b=\begin{pmatrix}20\\ 30\end{pmatrix}$$

$$B(1)=1, B(2)=2\Rightarrow Z=0, \pi^T=\begin{pmatrix}0 & 0\end{pmatrix}, A_B^{-1}=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}, A_B^{-1}\cdot b=\bar{b}=\begin{pmatrix}20\\ 30\end{pmatrix}$$

We commence with

$$\bar{c}_3=-4-0=-4\Rightarrow s=3\Rightarrow\lambda_0=\min\left\{\frac{20}{4},\frac{30}{3}\right\}=5\Rightarrow r=1$$

Example – Revised Simplex Method

$$\Rightarrow\lambda_0=\min\left\{\frac{20}{4},\frac{30}{3}\right\}=5\Rightarrow r=1\wedge B(1)=3, B(2)=2$$

Assuming $\tilde{B}$ as the former basis, we get $A_B^{-1}=P\cdot A_B^{-1}=\left(e^1\ \ldots\ \underbrace{p}_{\text{Column } r\ B^{-1}(s)}\ \ldots\ e^m\right)\cdot A_B^{-1}$, with $p=\begin{pmatrix}-\frac{\bar{a}_{1s}}{\bar{a}_{rs}}\\ \ldots\\ \frac{1}{\bar{a}_{rs}}\\ \ldots\\ -\frac{\bar{a}_{ms}}{\bar{a}_{rs}}\end{pmatrix}$

Thus, we have $A_B^{-1}=\begin{pmatrix}\frac{1}{4} & 0\\ -\frac{3}{4} & 1\end{pmatrix}\cdot\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}=\begin{pmatrix}\frac{1}{4} & 0\\ -\frac{3}{4} & 1\end{pmatrix}\Rightarrow\bar{b}=\begin{pmatrix}\frac{1}{4} & 0\\ -\frac{3}{4} & 1\end{pmatrix}\cdot b=\begin{pmatrix}\frac{1}{4} & 0\\ -\frac{3}{4} & 1\end{pmatrix}\cdot\begin{pmatrix}20\\ 30\end{pmatrix}=\begin{pmatrix}5\\ 15\end{pmatrix}$

$$\Rightarrow\pi^T=c_B^T\cdot A_B^{-1}=\begin{pmatrix}-4 & 0\end{pmatrix}\cdot\begin{pmatrix}\frac{1}{4} & 0\\ -\frac{3}{4} & 1\end{pmatrix}=\begin{pmatrix}-1 & 0\end{pmatrix}$$

Example – Revised Simplex Method

$$\bar{c}_1 = 0 - \begin{pmatrix} -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1, \bar{c}_2 = \bar{c}_3 = 0, \bar{c}_4 = -2 - \begin{pmatrix} -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \end{pmatrix} = -1, \bar{c}_5 = -3 - \begin{pmatrix} -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 3 \end{pmatrix} = -1 \Rightarrow s = 4$$

$$\Rightarrow \bar{a}^4 = A_B^{-1} \cdot a^4 = \begin{pmatrix} 1/4 & 0 \\ -3/4 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} 1/4 \\ 9/4 \end{pmatrix} \Rightarrow \lambda_0 = \min\left\{ \frac{5}{1/4}, \frac{15}{9/4} \right\} = \frac{60}{9} = \frac{20}{3}$$

$$\Rightarrow r = 2 \Rightarrow B(1) = 3, B(2) = 4 \Rightarrow$$

Thus, we have $A_B^{-1} = \begin{pmatrix} 1 & -\frac{1}{9} \\ 0 & \frac{4}{9} \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{4} & 0 \\ -\frac{3}{4} & 1 \end{pmatrix} = \begin{pmatrix} \frac{12}{36} & -\frac{1}{9} \\ -\frac{12}{36} & \frac{4}{9} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & -\frac{1}{9} \\ -\frac{1}{3} & \frac{4}{9} \end{pmatrix}$

$$\bar{b} = \begin{pmatrix} \frac{1}{3} & -\frac{1}{9} \\ -\frac{1}{3} & \frac{4}{9} \end{pmatrix} \cdot b = \begin{pmatrix} \frac{1}{3} & -\frac{1}{9} \\ -\frac{1}{3} & \frac{4}{9} \end{pmatrix} \cdot \begin{pmatrix} 20 \\ 30 \end{pmatrix} = \begin{pmatrix} 10/3 \\ 20/3 \end{pmatrix},$$

$$\pi^T = c_B^T \cdot \begin{pmatrix} \frac{1}{3} & -\frac{1}{9} \\ -\frac{1}{3} & \frac{4}{9} \end{pmatrix} = \begin{pmatrix} -4 & -2 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & -\frac{1}{9} \\ -\frac{1}{3} & \frac{4}{9} \end{pmatrix} = \begin{pmatrix} -2/3 & -4/9 \end{pmatrix}$$

Example – Revised Simplex Method

$$\bar{c}_1 = 0 - \begin{pmatrix} -2/3 & -4/9 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 2/3, \bar{c}_2 = 0 - \begin{pmatrix} -2/3 & -4/9 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 4/9, \bar{c}_3 = 0, \bar{c}_4 = 0,$$

$$\bar{c}_5 = -3 - \begin{pmatrix} -2/3 & -4/9 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 3 \end{pmatrix} = -1/3 \Rightarrow s = 5$$

$$\Rightarrow \bar{a}^5 = A_B^{-1} \cdot a^5 = \begin{pmatrix} 1/3 & -1/9 \\ -1/3 & 4/9 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 1/3 \\ 2/3 \end{pmatrix} \Rightarrow \lambda_0 = \min\left\{ \frac{10/3}{1/3}, \frac{20/3}{2/3} \right\} = 10$$

$$\Rightarrow r = 2 \Rightarrow B(1) = 3, B(2) = 5 \Rightarrow$$

Thus, we have $A_B^{-1} = \begin{pmatrix} 1 & -\frac{1}{2} \\ 0 & \frac{3}{2} \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & -\frac{1}{9} \\ -\frac{1}{3} & \frac{4}{9} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & -\frac{1}{3} \\ -\frac{1}{2} & \frac{2}{3} \end{pmatrix}$

$$\bar{b} = \begin{pmatrix} \frac{1}{2} & -\frac{1}{3} \\ -\frac{1}{2} & \frac{2}{3} \end{pmatrix} \cdot b = \begin{pmatrix} \frac{1}{2} & -\frac{1}{3} \\ -\frac{1}{2} & \frac{2}{3} \end{pmatrix} \cdot \begin{pmatrix} 20 \\ 30 \end{pmatrix} = \begin{pmatrix} 0 \\ 10 \end{pmatrix}, \pi^T = c_B^T \cdot \begin{pmatrix} \frac{1}{2} & -\frac{1}{3} \\ -\frac{1}{2} & \frac{2}{3} \end{pmatrix} = \begin{pmatrix} -4 & -3 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{2} & -\frac{1}{3} \\ -\frac{1}{2} & \frac{2}{3} \end{pmatrix} = \begin{pmatrix} -1/2 & -2/3 \end{pmatrix}$$

Example – Revised Simplex Method

$$\bar{c}_1 = 0 - \begin{pmatrix} -1/2 & -2/3 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1/2, \bar{c}_2 = 0 - \begin{pmatrix} -1/2 & -2/3 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 2/3, \bar{c}_3 = 0,$$

$$\bar{c}_4 = -2 - \begin{pmatrix} -1/2 & -2/3 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \frac{1}{2}, \bar{c}_5 = 0$$

$$\Rightarrow$$

$$x^T = (0,0,0,0,10) \wedge \pi^T = \begin{pmatrix} -1/2 & -2/3 \end{pmatrix} \text{ are optimal}$$

$$\Rightarrow c^T \cdot x = b^T \cdot \pi = 30$$

3.2 Analyzing the complexity

- Clearly, at first glance, you would assume that the main complexity effect of the revised simplex algorithm is from the fact that it's application reduces the values to be updated from an (m+1)x(n+1) tableau to a (m+1)x(m+1) matrix
- However, we have to generate the reduced costs by iteratively computing $\pi^T A_j$ for a considered non-basic column. Thus, if we have to do this for all these non-basic columns this requires $m \cdot (n - m)$ multiplications
- But this is not significantly less than the number of multiplications needed to update the tableau in the ordinary simplex method!

Positive effect of the revised simplex

- The practical complexity reducing effect of the revised simplex are somewhat more subtle but significant nonetheless
- First, it is quite likely that we need not compute the relative costs of every non-basic column if we (for instance) always take the first column with negative costs (improving alternative) (e.g. rule of Bland)
  - This significantly reduces on the average the computational effort to some fraction
  - This fraction is determined by the average number of columns that must be examined until one with negative relative cost is identified

Positive effect of the revised simplex

- Second, each pricing operation (i.e., the calculation of the reduced cost value $\pi^T A_j$ for a considered column j) uses the column A_j of the original Tableau
- We will see in many practical applications in Combinatorial Optimization these matrices are often very sparse (many zero entries)
  - Thus, the necessary prizing computations can be performed very efficiently
  - Moreover, the original matrix can be stored in a very compact way

Further refinement

- Bearing these ideas in mind, we can alternatively store the inverse basis matrix in **product form** and not as an (m+1)x(m+1) matrix.
- Each pivot operation can be represented as a multiplication with a matrix P that equals the (m+1)x(m+1) identity matrix except for column r that contains the vector

$$\eta = \begin{pmatrix} -x_{1,s}/x_{r,s} \\ -x_{2,s}/x_{r,s} \\ \dots \\ 1/x_{r,s} \\ \dots \\ -x_{m,s}/x_{r,s} \\ -x_{m+1,s}/x_{r,s} \end{pmatrix} \leftarrow r\text{th row}$$

Update at any stage *l*

- By using the current vector η, we can efficiently generate all matrices P_l and generate

$$CARRY^{(l)} = P_l \cdot P_{l-1} \cdot \dots \cdot P_1 \cdot CARRY^{(0)}$$

- Moreover, due to the current vector η, the matrix P_l can be stored very efficiently
- However, if the sequence of η becomes too long, it can be replaced by a shorter sequence
  - This replacement process is denoted as *reinversion*
  - It generates an equivalent but shorter sequence of pivots to attain the current basis
  - Such techniques can greatly reduce the storage and time required to perform the simplex algorithm, especially if special attention is paid to reducing the number of nonzero elements in the η-sequence (see Orchard-Hays (1968) or Lasdon (1970)).

3.3 Solving the Max Flow Problem

- In anticipation of the applications that are considered in the Sections 6 and 7 where we introduce and consider the Shortest Path Problem and the Max Flow Problem, in what follows, we show an interesting application of the revised simplex procedure
- Both applications are network problems that can be formulated as Linear Programs and the constraint matrix can be derived directly from the graph underlying the problem

The Max Flow Problem

3.3.1 Definition

Given a flow network $N = (s, t, V, E, b)$ with $n = |V|$ nodes and $m = |E|$ arcs, an instance of the Max Flow Problem (MFP) is defined as the optimization problem of finding a flow $f \in IR^m$ on each edge with maximal value v from the single source node s to the single terminal node t. On each edge, this flow has to be lower or equal to the capacity vector $b \in IR^m$.

Specific problem definition

- We now formulate this problem as an LP in a somewhat surprising way
- Since the arcs are numbered by $e_1,\ldots,e_m$, we introduce $C_1,\ldots,C_p$ as a complete enumeration of every chain (i.e., path) from *s* to *t*
- As known from the revised simplex algorithm introduced above, the theoretical Linear Program covers *m* rows and *p* columns but the considered LP in each step will comprise only *m* columns

Arc-chain incidence matrix $D=[d_{i,j}]$

- In order to unambiguously define possible paths, we determine the so-called arc-chain incidence matrix as follows:

$$\forall i \in \{1,\ldots,m\} : \forall j \in \{1,\ldots,p\} : d_{i,j} = \begin{cases} 1 & \text{if } e_i \in C_j \\ 0 & \text{otherwise} \end{cases}$$

- The capacity constraints are given by: $D \cdot f \le b$
- The objective function pursues the maximization of all the flows in all defined chains, i.e.,

$$\min c^T \cdot f$$

- with $c = (-1, \ldots, -1)$

Complete Linear program

- Thus, we obtain the following LP

$$\begin{aligned}&\min\ c^T \cdot f\\ &\text{s.t. } D \cdot f \le b \wedge f \ge 0\end{aligned}$$

- By introducing a slack vector $s \in IR^m$, we transform this LP into standard form and extend the respective vectors as follows

Flow vector: $\hat{f} = (f|s)$, cost vector $\hat{c} = (c = -1|0)$, and $\hat{D} = (D|E^m)$

- This leads to the LP

$$\begin{aligned}&\min\ z = \hat{c}^T \cdot \hat{f}\\ &\text{s.t. } \hat{D} \cdot \hat{f} = b \wedge \hat{f} \ge 0 \Rightarrow f \ge 0\end{aligned}$$

Consequences

- Clearly, each slack variable s_i represents the difference between the flow in arc *i* and the capacity b_i, with *i=1,…,m*
- We now apply the revised simplex algorithm
  - Fortunately, we do not have to apply the two-phase method since this problem is solvable with the trivial solution $f = 0, s = b$ that represents the zero flow
  - The criterion for a new column to enter the basis are negative reduced costs, given by

$$\bar{c}_j = c_j - \pi^T \cdot D_j < 0 \quad \Leftrightarrow \quad -1 - \pi^T \cdot D_j < 0 \Leftrightarrow -\pi^T \cdot D_j < 1$$

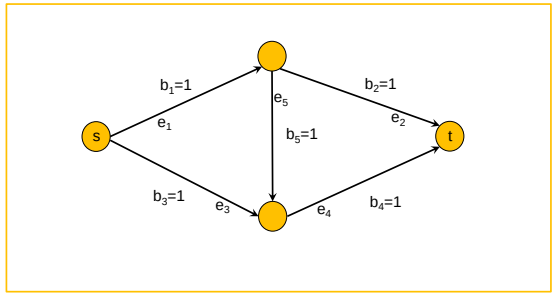
Since $c_j = -1$ for all structure variables

  - D_j is the *j*th column of the arc-chain incidence matrix *D*

Consequences

- Clearly, $-\pi^T \cdot D_j$ is the cost of the chain/path C_j under the weight vector $-\pi$
- In order to find a profitable column, we need to find the *shortest chain* from *s* to *t* under the weight vector $-\pi$ that weights less than 1
  - If that shortest chain/path, say C_j, has cost no less than 1, then the optimality criterion is satisfied
  - If not, we introduce C_j into the basis
- The calculation therefore requires only the maintenance of an *(m*+1)x*(m*+1) *CARRY* matrix and the repeated solution of the shortest-path problem

Max Flow Problem – Example



The matrix CARRY(0)

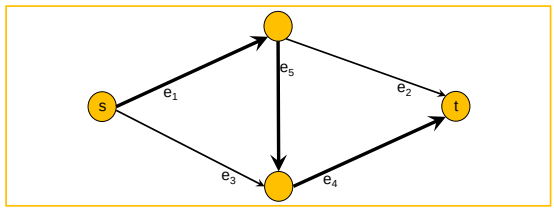
Dual solution $c_j - \pi^T \cdot A = 0 - \pi^T \cdot E = -\pi^T$

-z=0	0	0	0	0	0
s_1=1	1				
s_2=1		1			
s_3=1			1		
s_4=1				1	
s_5=1					1

- The initial dual solution is zero, i.e., π=0, and therefore, the initial weights are zero
- Hence, each path from s to t is minimal and has the length 0<1
- Therefore, it is a candidate to be integrated into the basis
- In what follows, we will determine a chain/path C_1

Max Flow Problem – path selection

- We introduce a shortest chain (of length zero) into the basis
- In order to complicate the computations a little bit, we start with a chain/path that is not in the optimal solution
- Specifically, we introduce the chain/path $C_1 = (e_1, \ e_5, \ e_4)$.
- This is illustrated below



Introducing C_1

- The corresponding column is $B^{-1} \cdot C_1 = C_1 =$

-1
1
0
0
1
1

- We update the current CARRY(1) matrix

Dual solution $c_j - \pi^T \cdot A = 0 - \pi^T \cdot E = -\pi^T$

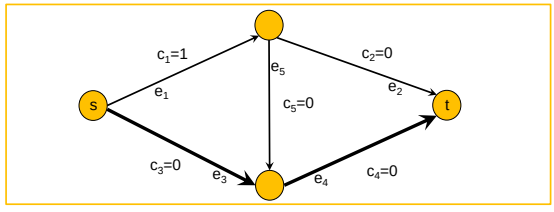
-z=0	0	0	0	0	0
s_1=1	1				
s_2=1		1			
s_3=1			1		
s_4=1				1	
s_5=1					1

⟹

-z=1	1	0	0	0	0
C_1=1	1				
s_2=1		1			
s_3=1			1		
s_4=0	-1			1	
s_5=0	-1				1

Max Flow Problem – path selection

- We again introduce a shortest chain (of length zero) into the basis
- Now, we introduce the chain/path $C_2 = (e_3 \quad e_4)$.
- This is illustrated below



Introducing C_2

- The column C_2 is

-1
0
0
1
1
0

and we have $B^{-1} \cdot C_2 =$

-1
0
0
1
1
0

$\bar{c}_j = -1 - \pi^T \cdot D_j$

- We update the current CARRY(2) matrix

Dual solution $c_j - \pi^T \cdot A = 0 - \pi^T \cdot E = -\pi^T$

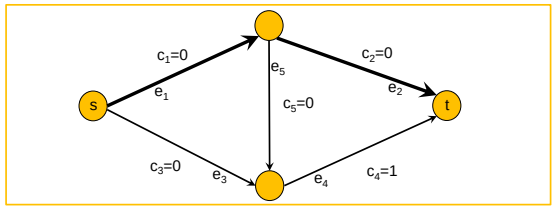
-z=1	1	0	0	0	0
C_1=1	1				
s_2=1		1			
s_3=1			1		
s_4=0	-1			1	
s_5=0	-1				1

⟹

-z=1	0	0	0	1	0
C_1=1	1				
s_2=1		1			
s_3=1	1		1	-1	
C_2=0	-1			1	
s_5=0	-1				1

Max Flow Problem – path selection

- We again introduce a shortest chain (of length zero) into the basis
- Now, we introduce the chain/path $C_3 = (e_1 \quad e_2)$.
- This is illustrated below



Introducing C_3

- The column C_3 is

-1
1
1
0
0
0

and we have $B^{-1} \cdot C_3 =$

-1
1
1
1
-1
-1

$\bar{c}_j = -1 - \pi^T \cdot D_j$

- We update the current CARRY(3) matrix

Dual solution $c_j - \pi^T \cdot A = 0 - \pi^T \cdot E = -\pi^T$

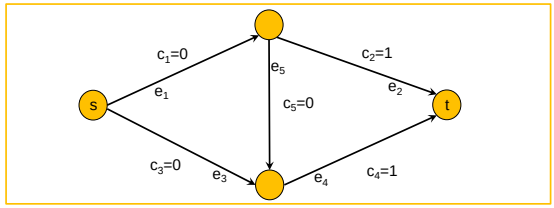
-z=1	0	0	0	1	0
C_1=1	1				
s_2=1		1			
s_3=1	1		1	-1	
C_2=0	-1			1	
s_5=0	-1				1

⇒

-z=2	0	1	0	1	0
C_1=0	1	-1			
C_3=1		1			
s_3=0	1	-1	1	-1	
C_2=1	-1	1		1	
s_5=1	-1	1			1

Max Flow Problem – path selection

- No zero chain/path exists between *s* and *t*
- Hence, an optimal solution with maximal flow 2 is found
- It comprises the paths $C_2 = (e_3 \quad e_4)$ and $C_3 = (e_1 \quad e_2)$



Alternatively introducing C_3

- The column C_3 is

-1
1
1
0
0
0

and we have $B^{-1} \cdot C_3 =$

-1
1
1
1
-1
-1

$\bar{c}_j = -1 - \pi^T \cdot D_j$

- We update the current CARRY(3) matrix

Dual solution $c_j - \pi^T \cdot A = 0 - \pi^T \cdot E = -\pi^T$

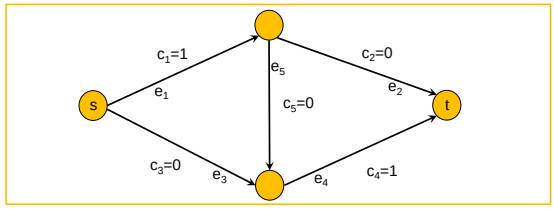
-z=1	0	0	0	1	0
C_1=1	1				
s_2=1		1			
s_3=1	1		1	-1	
C_2=0	-1			1	
s_5=0	-1				1

⇒

-z=2	1	0	0	1	0
C_3=1	1				
s_2=0	-1	1			
s_3=0			1	-1	
C_2=1				1	
s_5=1					1

Max Flow Problem – path selection

- No zero chain/path exists between *s* and *t*
- Hence, an optimal solution with maximal flow 2 is found
- It comprises the paths $C_2 = (e_3 \quad e_4)$ and $C_3 = (e_1 \quad e_2)$



3.4 The Dantzig-Wolfe Decomposition

- It often happens that a large linear program is actually a collection of smaller linear programs that are largely independent of each other
- Therefore, the revised simplex algorithm allows us to decompose the entire problem into smaller master- and subproblems
- The "communication" between these problems can be directly derived from the revised simplex
- Thanks to this decomposition, problems that may become too large to be solved efficiently (e.g., due to space requirements) can be solved very fast

Structure of the LP

- We analyze the entire LP with two subproblems
- This leads to the following scheme of a matrix

n_1 columns	n_2 columns	
D	*F*	m_0 rows
A	*0*	m_1 rows
0	*B*	m_2 rows

- With variables $x \in IR^{n_1}$ corresponding to the first n_1 columns and $y \in IR^{n_2}$ corresponding to the next n_2 columns

The complete LP

$$\min z = c^T \cdot x + d^T \cdot y$$

s.t.

$$D \cdot x + F \cdot y = b_0$$

$$A \cdot x = b_1$$

$$B \cdot y = o_2$$

$$x, y \geq 0$$

We denote the m_0 equations the coupling equations

The succeeding sets of rows are the subproblems *A* and *B*

The subproblem *A*

- We consider the constraints of subproblem *A*, i.e.,

$$A \cdot x = b_1$$

$$x \geq 0$$

- By Theorem 1.3.19, we know that any feasible point in this subproblem can be written as a convex combination of edge points. We denote these points as $x_1, \ldots, x_p$, with

$$x = \sum_{j=1}^{p} \lambda_j \cdot x_j \text{ where } \forall j \in \{1,\ldots,p\} : \lambda_j \geq 0 \text{ and } \sum_{j=1}^{p} \lambda_j = 1$$

Analogously: The subproblem *B*

- We consider the constraints of subproblem *B*, i.e.,

$$B \cdot y = o_2$$
$$y \geq 0$$

- By Theorem 1.3.19, we know that any feasible point in this subproblem can be written as a convex combination of edge points. We denote these points as $y_1, \ldots, y_q$, with

$$y = \sum_{j=1}^{q} \mu_j \cdot y_j \text{ where } \forall j \in \{1,\ldots,q\} : \mu_j \geq 0 \text{ and } \sum_{j=1}^{q} \mu_j = 1$$

Modified structure of the LP

- We replace *x* and *y* by their derived representations
- This leads to the following modified scheme (in what follows, denoted as the master problem)

Variables			
$\lambda_1, \ldots, \lambda_p$	$\mu_1, \ldots, \mu_q$	*Right hand side*	*Dimensions*
$Dx_1, \ldots, Dx_p$	$Fy_1, \ldots, Fy_q$	b_0	m_0 rows
1,…,1	*0,…,0*	1	1 row
0,…,0	*1,…,1*	1	1 row

- Costs

$$\min z = \sum_{j=1}^{p} \lambda_j \cdot c^T \cdot x_j + \sum_{j=1}^{q} \mu_j \cdot d^T \cdot y_j$$

while $\lambda, \mu \geq 0$ are the new variables $\lambda \in IR^p \wedge \mu \in IR^q$

Consequences

- Due to the transformation, we may obtain an astronomical number of columns, one for each vertex of each of the two subproblems
- However, the number of rows is significantly reduced from $m_0 + m_1 + m_2$ to m_0 +2
- Furthermore, the revised simplex method can be applied with a *CARRY* matrix of size $(m_0 + 3) \times (m_0 + 3)$
- Hence, much larger instances may fit into fast-access storage
  - Size of CARRY: $(m_0 + 3) \times (m_0 + 3)$ instead of $(m_0 + m_1 + m_2 + 1) \times (m_0 + m_1 + m_2 + 1)$
  - If, for instance, it holds that $m_0 = m_1 = m_2 = m$, we have $(m + 3)^2$ instead of $(3 \cdot m + 1)^2$, i.e., almost a factor of 3^2=9

Applying the revised simplex

- In the first row, we have the prices (reduced costs)
- This is a partitioned vector (π, α, β) where $\pi \in IR^{m_0}$ corresponds to the first m_0 rows and $\alpha, \beta \in IR$ to the next two rows of the master problem, respectively
- The reduced costs of the λ_j-column are given by

$$\bar{c}_j = c^T \cdot x_j - (\pi, \alpha, \beta) \cdot \begin{bmatrix} D \cdot x_j \\ 1 \\ 0 \end{bmatrix} = c^T \cdot x_j - \pi^T \cdot D \cdot x_j - \alpha$$

- Hence, the criterion for a column to be brought into the current basis is

$$c^T \cdot x_j - \pi^T \cdot D \cdot x_j - \alpha < 0 \Leftrightarrow c^T \cdot x_j - \pi^T \cdot D \cdot x_j < \alpha$$

Pricing problem *A*

- If $c^T \cdot x_j - \pi^T \cdot D \cdot x_j < \alpha$ holds for any vertex of Subproblem *A*, we have found a profitable column among the first *p* columns
- However, since an exhaustive exploration of the total number of possible columns (i.e., for each possible vertex of Subproblem *A*) would be impractical, we pursue the finding of the optimal vertex of the following LP

$$\min_{\text{all vertices } x_j \text{ of Subproblem } A} c^T \cdot x_j - \pi^T \cdot D \cdot x_j = \left(c^T - \pi^T \cdot D\right) \cdot x_j$$

- Thus, the pricing problem is

$$\begin{aligned} &\min \left(c^T - \pi^T \cdot D\right) \cdot x \\ &\text{s.t. } A \cdot x = b_1 \wedge x \geq 0 \end{aligned}$$

Pricing problem *A*

- If the objective function value of the optimal solution is lower than α the resulting column can be introduced into the basis
  - Hence, the master problem integrates the new column into the basis and, therefore, updates the current solution
  - The newly attained solution results in a new price vector sent to subproblem *A*
- Otherwise, no further improvement is possible and the solution of the master problem is kept unchanged according to the input of Subproblem *A*

Pricing problem *B*

- Similarly, we can determine if there is a favorable column among the last *q* columns by solving

$$\min \left(d^T - \pi^T \cdot F\right) \cdot y$$
$$\text{s.t. } B \cdot y = b_2 \wedge y \geq 0$$

- and comparing the attained minimum cost to *β*, by an argument analogous to that surrounding the pricing problem *A*

Procedure decomposition

opt := 'no'
set up CARRY with zero row *(π, α, β)*;
while *opt* = 'no' do
begin solve the LP
  min $(c^T - \pi^T D) \cdot x = z_0$ s.t. $Ax = b_1, x \geq 0$
  if $Z_0 < \alpha$
    then generate the column corresponding to the solution with this cost, and pivot in CARRY
    else begin solve the LP
      min $(d^T - \pi^T F) \cdot y = z_0$ s.t. $By = b_2, y \geq 0$
      if $Z_0 < \beta$
        then generate the column corresponding to the solution with this cost, and pivot in CARRY
        else *opt* := 'yes'
    end
end

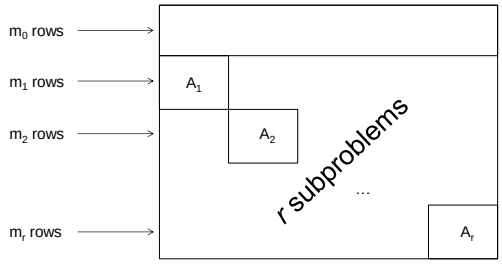
Summary

We verbally sum up the operations of the decomposition method:

- After being solved, the master problem, based on its overall view of the entire situation, sends a price to subproblem *A*
- This subproblem *A* then responds with a solution (a proposal) for possibly improving the overall problem, based on its local information and the price
- The master problem then weighs the cost (Z_0) of this proposal against its criterion *α* for subproblem *A*
  - If the proposal is cheaper than *α*, it is implemented by bringing it into the basis. This results in an update of the current solution (and prices)
  - If not, subproblem *B* is sent a price vector and asked for a proposal.
- As long as a subproblem can produce a favorable proposal, the master problem can find a favorable pivot
- When neither subproblem can come up with a favorable proposal, we have reached an optimal solution of the entire problem

Conclusions

- What we have done for two subproblems can clearly be done for any number of subproblems, in which case the constraint matrix will take the form shown below



An illustrative example

- A little arithmetic shows the effectiveness of the approach in size reduction of the mapped problems
- Suppose we have 100 subproblems, each with 100 rows, and 100 coupling equations. Then the original problem has 10,100 rows
  - *CARRY* has 10,101 x 10,101 entries, i.e., approximately 10^8 entries. May be too many for the working memory of a computer
  - The master problem has a *CARRY* with 201 x 201 entries. These are approximately 4 x 10^4 entries, which is practical to store in the fast memory of any reasonably large computer
- All in all, the clear advantage of the decomposition algorithm is its effect on the space requirements
- Note that it cannot be said much about the time requirements, because we do not know how many times the subproblems must be solved before optimality is reached

Additional literature to Section 3

- Beale, E.M.L. (1968): Mathematical Programming in Practice. John Wiley & Sons, Inc., New York.
- Dantzig, G.B., Orden, A., and Wolfe, P. (1955): The generalized simplex method for minimizing a linear form under linear inequality restraints. Pacific Journal of Mathematics 5, 183–195.
- Dantzig, G.B., Wolfe, P. (1960): "Decomposition Principle for Linear Programming". *Operations Research,* 8(1), 101-11.
- Dantzig, G.B. (1963): *Linear Programming and Extensions.* Princeton University Press, Princeton, N.J.
- Orchard/Hays, W. (1968): Advanced Linear-Programming Computing Techniques. McGraw-Hill Book Company, New York.
- Lasdon, L.S. (1970): Optimization Theory for Large Systems. London: MacMillan, Inc.